Cost-based Scheduling for Data-Intensive Applications on Global Grids

Srikumar Venugopal and Rajkumar Buyya
Grid Computing and Distributed Systems (GRIDS) Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne, Australia
Email:{srikumar,raj}@cs.mu.oz.au

Abstract

We present an algorithm for scheduling distributed data intensive Bag-of-Task applications on Data Grids that have costs associated with requesting, transferring and processing datasets. We evaluate the algorithm on a Data Grid testbed and present the results.

1 Introduction

Data Grids [2] have evolved to tackle the challenges of accessing, processing and managing large distributed datasets that are posed by distributed data-intensive applications in areas such as high-energy physics, astronomy and bioinformatics. In Data Grids, there can be a lot of pressure on the data infrastructure (i.e., network and storage elements). This can lead to overloading of resources and appearance of network "hot spots" as is commonly observed in the World Wide Web. Pricing resources to reflect supply and demand in order to regulate their usage has been explored in previous work [1]. In such a scenario, scheduling strategies will have to take into account resources of varying capabilities with varying execution, transfer and storage costs. While such strategies have been proposed and evaluated for computational grids, no study has yet been made for similar requirements in Data Grid environments.

In this paper, we introduce an algorithm for cost-based scheduling for a data-intensive Bag-of-Tasks(BoT) application on a Data Grid. Each of the tasks within the application depends on mutiple datasets that may be available from multiple data sources. The algorithm minimizes either the overall cost or the time of execution depending on the user's preference subject to two user-defined constraints - the deadline by which the processing must be completed and the overall budget for performing the computation. In doing so, it is guided by factors such as cost and speed of accessing, transferring and processing data.

2 Scheduling

Figure 1 shows a typical Data Grid environment which is composed of storage resources, or *data hosts*, which store

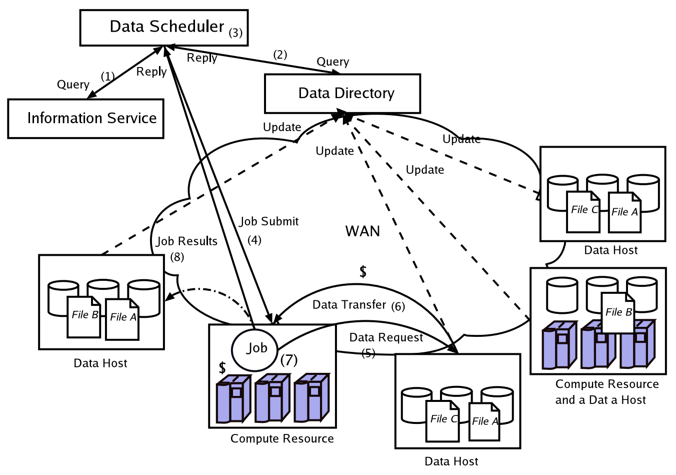


Figure 1. A cost-based Data Grid environment

the data and compute resources which run the jobs that execute upon the data. The datasets may be replicated at various sites within this data grid depending on the policies set by the administrators of the storage resources and/or the producers of data. The scheduler is able to query a data directory for information about the locations of the datasets and their replicas. We associate economic costs with the access, transfer and processing of data. The processing cost is levied upon by the computational service provider, while the transfer cost comes on account of the access cost for the data host and the cost of transferring datasets from the data host to the compute resource through the network.

A job (equivalent to a task in a BoT) is the atomic unit of computation within this model. Each job requires one or more datasets as input. Each dataset is available through one or more data hosts. The steps for submitting a job to the grid shown in Fig. 1 are as follows: The scheduler gathers information about the available compute resources (1) and about the datasets and data hosts (2). It then makes decision on where to submit the job (3). The job is dispatched to the selected remote compute resource (4) where it requests for the datasets from the replica locations selected by the scheduler (5 & 6). After the job has finished processing (7), the results are sent back to the scheduler host or another storage resource(8). This process is repeated for all the jobs within the BoT.

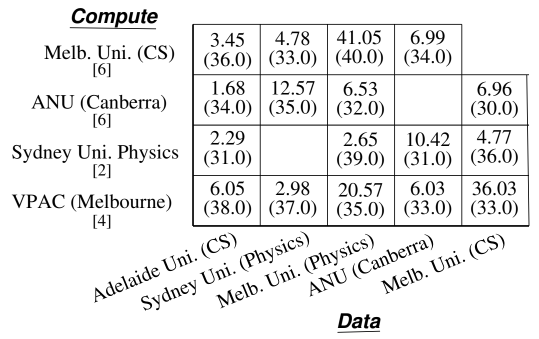
We have two objective functions here, either to produce the least expensive execution within deadline or to finish the execution in the least time possible within the budget. A detailed listing of the proposed algorithm is given in [3]. The algorithm is in two parts described below:

Map jobs to resources: For each job, we build a *resource set* that consists of one compute resource for executing the job and one data resource for each dataset requested by the job. We iterate through the list of datasets requested by the job. For each dataset, we pick the combination of a compute resource and a data host that returns the lowest value for expected transfer cost or time depending on the minimization. Then we see if our choice of compute resource is better when the data hosts selected for previous datasets are considered. This procedure ensures that the choice of the compute resource and the resource set so formed at the end of each iteration is better than those selected in all previous iterations.

Dispatch jobs: In the dispatching section, starting with the job with the least cost or least execution time, we submit the jobs to the compute resources selected for them in the mapping step if the allocation for the resources has not been exhausted. We also check if the deadline or budget constraints are being violated while submitting a job.

3 Experiments and Results

We have implemented the scheduling algorithm presented in Section 2 within the Gridbus Broker. The experimental setup used in our evaluation is summarized in Figure 2. It consists of an Australia-wide Data Grid consisting of IBM eServer machines in Melbourne, Adelaide, Sydney and Canberra along with a cluster and a PC in Melbourne. Each of these played the roles of a compute resource or a data host or both and were assigned different usage costs. We also assigned costs to the network links between these machines and took into consideration the available bandwidth while scheduling. Table 1 shows the summary of the results that were obtained for a set of 125 jobs with a 2 hour deadline and budget of 500,000 G$. The average costs per job incurred during cost and time minimization are 562.6 G$ and 959 G$ with standard deviations of 113 and 115 respectively. Mean wall clock time taken per job (including computation and data transfer time) was 167 secs for cost minimization and 135 secs for time minimization with standard deviations 16.7 and 19 respectively. As expected, cost minimization scheduling produces minimum computation and data transfer expenses whereas time minimization completes the experiments in the least time. A detailed discussion about the experimental results in given in [3].



Compute \ Data	Adelaide Uni. (CS)	Sydney Uni. (Physics)	Melb. Uni. (Physics)	ANU (Canberra)	Melb. Uni. (CS)
Melb. Uni. (CS) [6]	3.45 (36.0)	4.78 (33.0)	41.05 (40.0)	6.99 (34.0)	
ANU (Canberra) [6]	1.68 (34.0)	12.57 (35.0)	6.53 (32.0)		6.96 (30.0)
Sydney Uni. Physics [2]	2.29 (31.0)		2.65 (39.0)	10.42 (31.0)	4.77 (36.0)
VPAC (Melbourne) [4]	6.05 (38.0)	2.98 (37.0)	20.57 (35.0)	6.03 (33.0)	36.03 (33.0)

Figure 2. Testbed used in evaluation. The numbers without paranthesis in each cell represent average available bandwidth of link while those within are the costs assigned to links. The square brackets represent cost of execution per second on that resource. All costs are in Grid Dollars(G$)

Table 1. Summary of Evaluation Results

Minimization	Total Time(min)	Compute Cost(G$)	Data Cost(G$)	Total Cost(G$)
Cost	80	31198.27	39126.65	70324.93
Time	54	76054.90	43821.64	119876.55

4 Conclusion and Future Work

We have presented here an algorithm for executing jobs on data grids which takes into account both cost of processing and of data transfer and user-defined constraints such as deadline and budget. The algorithm explicitly deals with jobs that require multiple datasets from multiple data sources. We have presented empirical results obtained from evaluating the algorithm on a Data Grid testbed. We plan to conduct further evaluations with a testbed with various degrees of replication to conclusively state that the algorithm minimizes its objective functions

References

[1] R. Buyya et al. A Case for Economy Grid Architecture for Service-Oriented Grid Computing. In *Proc. of 10th IEEE International Heterogeneous Computing Workshop (HCW 2001)*, San Francisco, California, USA, April 2001.

[2] A. Chervenak et al. The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets. *Journal of Network and Computer Applications*, 23(3):187–200, 2000.

[3] S. Venugopal and R. Buyya. An economy-based algorithm for scheduling data-intensive applications on global grids. Technical Report GRIDS-TR-2004-11, GRIDS Laboratory, University of Melbourne, Australia, Dec 2004.